Attorney Docket No. ROC920030366US1
Confirmation No. 7130

PATENT

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Ex parte Paul Reuben Day, Brian Robert Muras, Anne Marie Ryg

Appeal No. ___________
Application No. 10/754,010

APPEAL BRIEF

Attorney Docket No. ROC920030366US1 PATENT
Confirmation No. 7130

CERTIFICATE OF ELECTRONIC TRANSMISSION
I hereby certify that this correspondence for Application No. 10/754,010 is being electronically transmitted to Technology Center 2167 via EFS-WEB, on August 17, 2009.

/Scott A. Stinebruner/ August 17, 2009
Scott A. Stinebruner, Reg. No. 38,323 Date

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Paul Reuben Day et al.	Art Unit:	2167
Application No.:	10/754,010	Examiner:	Kimberly M. Lovel
Filed:	January 8, 2004		
For:	METHOD AND SYSTEM FOR A SELF-HEALING QUERY ACCESS PLAN		

Mail Stop Appeal Brief - Patents
Commissioner for Patent
P.O. Box 1450
Alexandria, VA 22213-1450

APPEAL BRIEF

I. REAL PARTY IN INTEREST

This application is assigned to International Business Machines Corporation, of Armonk, New York.

II. RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences.

III. STATUS OF CLAIMS

Claims 1, 3-8 and 10-12 are pending in the Application, stand rejected, and are now on appeal. Claims 2, 9 and 13-21 have been canceled without prejudice. Applicant previously appealed the rejection of claims 1, 3-8, and 10-12, filing an appeal brief on December 9, 2008.

Subsequently, the Examiner reopened prosecution and rejected claims 1, 3-8 and 10-12 in the Non-Final Office Action of March 17, 2009. Applicant has reinstated the appeal and will address the new rejections made in the Non-Final Office Action of March 17, 2009.

IV. STATUS OF AMENDMENTS

There have been no amendments filed subsequent to the non-final rejection mailed March 17, 2009.

V. SUMMARY OF CLAIMED SUBJECT MATTER

Applicant's invention is generally directed to a database engine and optimizer framework that support the ability to automatically respond to execution errors that occur during execution of a query plan to allow for continued execution of a query plan. (Application, page 6, lines 2-4).

Specifically, conventional database management systems rely on query optimizers to choose the most efficient way to execute a database query. The output of a query optimizer is typically referred to as an "execution plan," "access plan," or just "plan," which typically incorporates low-level information telling the database engine that ultimately handles a query precisely what steps to take (and in what order) to execute the query. (Application, page 2, lines 20-24). Due to the fact that number of possibly query forms that can be used to represent a given query, and due to the need to respond relatively quickly to queries, a query optimizer often has only a limited amount of time to pare the search space of possible execution plans down to an optimal plan for a particular query. (Application, page 3, lines 3-16). As a result, in many instances a query optimizer will not generate the perfect plan by which to execute a query.

In some instances, a query plan may encounter errors during execution, some of which may be so significant as to require termination of execution of the query. These errors, which are termed execution errors, and which may also be referred to as function checks, cause the execution of a query to be halted when they are encountered. Conventional database engines include built-in error reporting and error handling facilities, and will typically report an error that includes some type of identifying code as well as the location in the query where the error occurred. (Application, page 3, lines 17-21). However, these engines do not have any ability to repair or correct the error without manual intervention. As a result, in response to such an error, the next step for the database user typically involves calling a customer support engineer and trying to resolve the problem via telephone or e-mail, leading to user frustration, system downtime, and lower productivity.

In contrast, embodiments consistent with the invention attempt to minimize the impact of execution errors that occur during the processing of the query plan for a database query through the use of a self-healing database engine and optimizer framework that support automatically responding to execution errors to allow continued execution of a query plan. (Application, page 4, lines 2-4). Upon encountering an execution error of the type that ordinarily halts execution of a query plan, the database engine automatically initiates a rebuilding of the query plan and executes the rebuilt query plan. (Application, page 4, lines 4-6). In some embodiments, if an error is encountered in the rebuilt query plan then the query implementation methods are analyzed, and if a query function is identified for which an alternative implementation method is available, then this alternative implementation method is substituted to create a new query plan. The new query plan is then executed to determine if the error is corrected. (Application, page 4, lines 6-10).

For the convenience of the Board, independent claims 1, 7 and 12 have been reproduced below and annotated with references to the specification and drawings to satisfy the requirement to concisely explain the claimed subject matter:

Independent Claim 1

A method (Application, Fig. 3, page 9, line 27 to page 12, line 9) for automatic handling of errors within a database engine (Application, Fig. 2, block 44, page 8, lines 14-15), the method comprising the steps of:

detecting an error while executing a query access plan (Application, Fig. 2, block 50, Fig. 3, block 304, page 8, lines 12-14, page 10, lines 4-6), wherein the error is an execution error of a type that halts execution of the query access plan (Application, page 10, lines 4-6), and wherein the query access plan is of the type generated by a query optimizer (Application, Fig. 2, blocks 42, 50, page 8, lines 10-14);

in response to detecting the error, automatically rebuilding the query access plan with the query optimizer to generate a new query access plan (Application, Fig. 3, block 308, page 10, lines 13-23); and

executing the new query access plan to generate at least a portion of a result set for storage or display (Application, Fig. 3, blocks 310, 312, page 10, lines 24-27).

Independent Claim 7

A method (Application, Fig. 3, page 9, line 27 to page 12, line 9) for automatic handling of errors within a database engine (Application, Fig. 2, block 44, page 8, lines 14-15), the method comprising the steps of:

receiving an error while executing a function within a query access plan (Application, Fig. 2, block 50, Fig. 3, block 304, page 8, lines 12-14, page 10, lines 4-6, page 11, lines 18-20), wherein the error is an execution error of a type that halts execution of the query access plan (Application, page 10, lines 4-6), and wherein the query access plan is of the type generated by a query optimizer (Application, Fig. 2, blocks 42, 50, page 8, lines 10-14);

identifying a first implementation method of the function within the query access plan (Application, Fig. 3, blocks 316-318, page 11, lines 18-23);

rebuilding the query access plan with the query optimizer by replacing the first implementation method with a second implementation method of the function so as to generate a new query access plan (Application, Fig. 3, block 318, page 11, lines 20-25); and

executing the new query access plan to generate at least a portion of a result set for storage or display (Application, Fig. 3, blocks 310, 312, page 10, lines 24-27, page 12, lines 3-7).

Independent Claim 12

A method (Application, Fig. 3, page 9, line 27 to page 12, line 9) for automatic handling of errors within a database engine (Application, Fig. 2, block 44, page 8, lines 14-15), the method comprising the steps of:

executing a query access plan (Application, Fig. 2, block 50, Fig. 3, block 302, page 10, lines 1-2) comprising a plurality of functions (Application, page 11, lines 6-17, page 12, lines 10-26), each function including a first implementation method (Application, page 11, lines 11-23), and the query access plan of the type generated by a query optimizer (Application, Fig. 2, blocks 42, 50, page 8, lines 10-14);

detecting a first error when executing a first function (Application, Fig. 3, block 304, page 10, lines 4-6), wherein the first error is an execution error of a type that halts execution of the query access plan (Application, page 10, lines 4-6);

rebuilding the query access plan with the query optimizer to generate a new query access plan (Application, Fig. 3, block 308, page 10, lines 13-23);

executing the new query access plan to generate at least a portion of a result set for storage or display (Application, Fig. 3, blocks 310, 312, page 10, lines 24-27);

receiving a second error while executing the first function within the new query access plan (Application, Fig. 3, block 316, page 11, lines 18-20); and

rebuilding the new query access plan by replacing the first implementation method with a second implementation method of the function (Application, Fig. 3, block 318, page 11, lines 20-25).

Other support for the claimed subject matter may generally be found in Fig. 3 and the accompanying text at pages 9-14 of the Application as filed. In addition, it should be noted that, as none of the claims recite any means plus function or step plus function elements, no identification of such elements is required pursuant to 37 CFR §41.37(c)(1)(v). Furthermore, there is no requirement in 37 CFR §41.37(c)(1)(v) to provide support for the subject matter in the separately argued dependent claims, as none of these claims recite means plus function or step plus function elements, and so no discussion of any of these claims is provided.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

A. Claims 1, 3 and 6 are rejected under 35 U.S.C. § 103 (a) as being unpatentable over the article "Efficient Mid-Query Re-Optimization of Sub-Optimal Query Execution Plans" by Kabra et al. (hereinafter "Kabra") in view of Brundage et al. (USPG Pub. No. 2004/0267760) (hereinafter Brundage").

B. Claims 4, 5, 7, 8, and 10-12 are rejected to under 35 U.S.C. § 103 (a) as being unpatentable over the article by Kabra in view of Brundage and further in view of Lohman et al. (USPG Pub. No. 2002/0198867) (hereinafter "Lohman").

VII. ARGUMENT

Applicant respectfully submits that the Examiner's rejections of claims 1, 3-8 and 10-12 are not supported on the record, and should be reversed.

A. Claims 1, 3 and 6 are not unpatentable over Kabra in view of Brundage

Claims 1, 3, and 6 stand rejected as being unpatentable over Kabra in view of Brundage. Applicant respectfully submits, however, that in the instant case, the Examiner has failed to establish a *prima facie* case of obviousness as to claims 1, 3 and 6, and thus, the rejections thereof should be reversed.

Based on the Supreme Court's decision in KSR, 127 S. Ct. at 1734, a *prima facie* showing of obviousness still requires that the Examiner establish that the differences between a claimed invention and the prior art "are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art." 35 U.S.C. §103(a). Such a showing requires that all claimed features be disclosed or suggested by the prior art. Four factors generally control an obviousness inquiry: 1) the scope and content of the prior art; 2) the differences between the prior art and the claims; 3) the level of ordinary skill in the pertinent art; and 4) secondary considerations of non-obviousness, such as commercial success of products covered by the patent claims, a long felt but unresolved need for the invention, and failed attempts by others to make the invention. KSR, 127 S. Ct. at 1734 (quoting Graham v. John Deere Company, 383 U.S. 1, 17-18 (1966)) ("While the sequence of these questions might be reordered in any particular case, the [Graham] factors continue to define the inquiry that controls.").

Moreover, in KSR, the Court explained that "[o]ften, it will be necessary for a court to look to interrelated teachings of multiple patents; the effects of demands known to the design community or present in the marketplace; and the background knowledge possessed by a person having ordinary skill in the art, all in order to determine whether there was an apparent reason to combine the known elements in the fashion claimed by the patent at issue" and "[t]o facilitate review, this analysis should be made explicit." KSR, 127 S. Ct. at 1740-41 *quoting* In re Kahn, 441 F.3d 977, 988, 78 USPQ2d 1329, 1336 (Fed. Cir. 2006) ("[R]ejections on obviousness grounds cannot be sustained by mere conclusory statements; instead, there must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness."). But, not every combination is obvious "because inventions in most, if not all, instances rely upon building blocks long since uncovered, and claimed discoveries almost of necessity will be combinations of what, in some sense, is already known." KSR, 127 S. Ct. at 1741.

As a result, after KSR, while there is no rigid requirement for an explicit "teaching, suggestion or motivation" to combine references, there still must be some evidence of "a reason that would have prompted a person of ordinary skill in the relevant field to combine the elements in the way the claimed new invention does" in an obviousness determination. KSR, 127 S. Ct. at 1731.

Applicant respectfully submits that, in the instant case, the Examiner has failed to establish a *prima facie* case of obviousness as to claims 1, 3, and 6, and as such, the rejections thereof should be reversed. Applicant's remarks in rebuttal to the Examiner's rejections are presented below, starting with the relevant independent claims and followed by a discussion of selected dependent claims. In some cases, specific discussions of particular claims are not made in the interest of streamlining the appeal. The omission of a discussion with respect to any particular claim, however, should not be interpreted as an acquiescence as to the merits of the Examiner's rejection of the claim, particularly with respect to claims reciting features that are addressed in connection with the rejection applied to other claims pending in the appeal.

Independent Claim 1

Claim 1 recites a method for automatic handling of errors within a database engine, which includes detecting an error while executing a query access plan; in response to detecting the error, automatically rebuilding the query access plan with the query optimizer to generate a new query access plan; and executing the new query access plan to generate at least a portion of a result set for storage or display. The claim also recites that the error is an execution error of a type that halts execution of the query access plan, and that the query access plan is of the type generated by a query optimizer.

Notably, the error that is addressed in claim 1 is specifically an execution error of a type that halts execution of the query access plan. As described in page 10, lines 1-6 of the Application, during execution, the database engine 44 may, in step 304, detect an error that halts execution of the query. Within some database environments, such an error is known as a function check. A number of different such errors may be detected; some relate to opening various tables; some relate to formatting the result set; some relate to problems with selection criteria; and many other types of errors are possible as well. Moreover, as indicated on page 12, lines 21-23 of the Application, execution errors sometimes occur because of problems in the database engine 44 or the optimizer 42, with a "patch" typically issued from the database vendor that addresses the problems. Nonetheless, the claimed execution error halts execution of a query, and is generally a type of involuntary operation that essentially halts a query and halts its continued execution.

In rejecting claim 1, the Examiner relies on Kabra and Brundage. The Examiner asserts that Kabra discloses limitations of the claim 1 at the abstract, lines 6-8 and page 109, col. 2, line 34 to page 110, col. 1, line 15. The Examiner admits, however, that Kabra fails to explicitly disclose the limitation "wherein the error is an execution error of a type that halts execution of the query access plan," required by claim 1. For this, the Examiner now cites Brundage, and specifically paragraphs [0047], [0183], [0185] and [0220] thereof.

Applicant again agrees with the Examiner that Kabra does not disclose or suggest each limitation of claim 1. For example, as previously stated, Kabra generally discloses improving the optimization of database queries by detecting a sub-optimal query and perhaps modifying or re-optimizing the query, but the decision to perform such a modification or optimization is made by the optimizer, and is more or less a voluntary operation performed on the basis of collected performance statistics. Execution errors that halt execution of a query, on the other hand, are involuntary operations that essentially terminate a query and prohibit its continued execution. And although the Examiner's new reference Brundage does in fact disclose the concept of a fatal error (see, e.g., "the error is fatal (re: terminate execution)" at paragraph [0220] of Brundage), Applicant respectfully submits that the combination of Kabra and Brundage still does not disclose or suggest all of the limitations in the same manner claimed.

First, the mere mention in Brundage of the concept of a fatal error adds little to the rejection. Applicant has already acknowledged that execution errors of the type that halt execution of a query are known in the Background portion of the specification (see, e.g., page 3, lines 17-21 of the Application). As discussed in Section V above, such errors, which are termed execution errors in the specification, and which may also be referred to as function checks, cause the execution of a query to be halted when they are encountered. Brundage's disclosure of these types of errors therefore adds little to the known state of the art as it pertains to claim 1.

Applicant has also noted that conventional database engines include built-in error reporting and error handling facilities, and will typically report an error that includes some type of identifying code as well as the location in the query where the error occurred. (Application, page 3, lines 17-21). However, these engines do not have any ability to repair or correct the error without manual intervention. As a result, in response to such an error, the next step for the database user typically involves calling a customer support engineer and trying to resolve the problem via telephone or e-mail, leading to user frustration, system downtime, and lower productivity. Brundage, despite disclosing the general concept of a fatal error, does not disclose or suggest any manner of handling such errors that is different than that which was already acknowledged by Applicants in the Application, and certainly does not disclose any method of handling execution errors by automatically rebuilding and executing a query access plan in response to detecting such an error, as required by claim 1.

Claim 1, in particular, recites detecting an error of a type that halts execution of the query access plan while executing a query access plan, and **in response to detecting the error automatically rebuilding the query access plan** with the query optimizer to generate a new query access plan, and **executing the new query access plan** to generate at least a portion of a result set. Brundage does not disclose or suggest any functionality capable of catching errors that halt execution and then handling such errors by recompiling and continuing with query execution. In addition, with respect to Kabra, Applicant submits that execution errors are entirely different in kind from the types of "errors" contemplated in Kabra (sub-optimally executing queries), so it is not a mere obvious extension of Kabra to apply its disclosed method to address execution errors.

Therefore, combining Brundage with Kabra, which merely discloses detection of a sub-optimal query and voluntary modification or re-optimization of the same (and thus operating only on events that are not analogous to execution-type (fatal) errors), the proposed combination does not disclose or suggest any type of error handling facility for execution-type (fatal) errors that avoids having to halt execution of a query by automatically rebuilding and executing the access plan for the query. Applicant therefore respectfully submits that the combination of Kabra and Brundage still does not disclose each and every limitation of claim 1.

Second, it is only through the benefit of hindsight and through the prism of Applicant's disclosure that the Examiner can extrapolate that the combination of Kabra and Brundage discloses or suggests claim 1. In particular, Applicant respectfully disagrees with the Examiner's argument at page 5 that Kabra and Brundage are combinable, namely, "[i]t would have been obvious to one of ordinary skill in the art to utilize Kabra's method to re-optimize a sub-optimal query plan to re-optimize the query of Brundage after the fatal error…to improve the performance and efficiency of applications through the generation of optimal plans." There is nothing in either reference that would suggest to one of ordinary skill in the art that Kabra's re-optimization method could be used to address execution-type errors. Kabra, as admitted by the Examiner, does not suggest any applicability of its method to anything other than sub-optimally executing queries. Furthermore, the mere mention of fatal errors in Brundage, without any disclosure whatsoever regarding how such errors are handled, does little or nothing to address this shortcoming in Kabra. In fact, Applicant can find no particular relevance of Brundage to the claim beyond a simple keyword match with the term "fatal error" since the reference is otherwise unconcerned with error handling. There must be some reasonable expectation that one of ordinary skill in the art would be motivated to use Kabra's optimization method to address instances where a query execution must be halted as the result of encountering an execution error, and neither Kabra nor Brundage provides any objective evidence of any such motivation. In fact, neither reference even addresses how execution errors of the type recited in claim 1 should be handled. As noted above, the differences between a sub-optimal query (which does not require the query to be halted) and an execution error (which conventionally requires execution to be halted) are such that it is not a mere obvious modification to retrofit the Kabra process to address execution errors in the same way as sub-optimal queries are addressed. The Kabra process, for example, optimizes sub-optimal queries by reallocating system resources or revising a query access plan for improved performance (e.g., by revising a join order or table access method). There has been no evidence proffered that there would be any reasonable expectation by one of ordinary skill in the art that such operations would address execution errors that would otherwise completely halt execution of a query.

Therefore, given the fundamental principles behind the respective implementations of the references and Applicant's claim 1 are so different, it would only be through the benefit of Applicant's disclosure that one of ordinary skill in the art would ever contemplate taking the disparate pieces from the Kabra and Brundage designs and combining them together in the manner suggested by the Examiner. Such is the essence of hindsight-based analysis, and is antithetical to the requirements for establishing a *prima facie* case of obviousness.

Thus, Applicant respectfully submits that claim 1 is non-obvious over Kabra and Brundage. Furthermore, given the caution against participating in piecemeal examination in MPEP 707.07(g), Applicant submits that claim 1 is in condition for allowance at this time. Reversal of the Examiner's rejection of claim 1, and allowance of that claim as well as of claims 3-6 which depend therefrom, are therefore respectfully requested.

Dependent Claim 3

Claim 3 depends from claim 1 and additionally recites that the error is a function check. The Examiner cites Kabra at page 109, column 2, lines 29-33 and Brundage at paragraphs [0183], [0185], and [0220] for allegedly disclosing this feature. However, the cited disclosure of Kabra at page 109 does not disclose a function check, which is recognized in the art, and specifically defined in the specification, as a type of error that halts execution in a database engine. Moreover, the Examiner's assertion that Kabra discloses a function check is completely inconsistent with the Examiner's admission (made in connection with the rejections of claims 1, 7 and 12) that Kabra does not disclose an error that halts execution of a query plan. In addition, while Brundage discloses "fatal errors," Brundage does not specifically disclose the concept of a "function check," which by virtue of claim differentiation, is required to be a specific type of "execution error of a type that halts execution of the query access plan."

Applicant therefore respectfully submits that when claim 3 is read in context with the limitations of claim 1 from which it depends, the Examiner's rejection based on the combination of Kabra and Brundage cannot be sustained. Reversal of the Examiner's rejection of claim 3, and allowance of the claim, are therefore respectfully requested for these reasons and the reasons set out in connection with claim 1.

Dependent Claim 6

Claim 6 depends from claim 1 and additionally recites reporting the error. The Examiner cites Brundage at paragraph [0220] for allegedly disclosing this feature. However, as argued above in connection with claim 1, Applicant respectfully submits that when claim 6 is read in context with the limitations of claim 1 from which it depends, the Examiner's rejection based on the combination of Kabra and Brundage cannot be sustained despite the disclosure in Brundage that "the first must be a string sequence, and describes a message to the user." Reversal of the Examiner's rejection of claim 6, and allowance of the claim, are therefore respectfully requested for these reasons and the reasons set out in connection with claim 1.

B. <u>Claims 4, 5, 7, 8 and 10-12 are not unpatentable over Kabra in view of Brundage and further in view of Lohman</u>

Claims 4, 5, 7, 8 and 10-12 stand rejected as being unpatentable over Kabra in view of Brundage and further in view of Lohman. Applicant respectfully submits, however, that in the instant case, the Examiner has failed to establish a *prima facie* case of obviousness as to claims 4, 5, 7, 8, and 10-12 and thus, the rejections thereof should be reversed. For simplicity, Applicant will first discuss the independent claims.

<u>Independent Claim 7</u>

Claim 7 generally recites a method for automatic handling of errors within a database engine. This method includes receiving an error while executing a function within a query access plan. The error is an execution error of a type that halts execution of the query access plan, and the query access plan is of the type generated by a query optimizer. The method also includes identifying a first implementation method of the function within the query access plan, rebuilding the query access plan with the query optimizer by replacing the first implementation method with a second implementation method of the function so as to generate a new query access plan, and executing the new query access plan to generate at least a portion of a result set for storage or display.

In rejecting claim 7, the Examiner also relies on the same arguments and citations of Kabra and Brundage made in connection with claim 1. The Examiner admits, however, that the combination of Kabra and Brundage does not disclose "identifying a first implementation method of the function within the new query access plan" and "rebuilding the new query access plan by replacing the first implementation method with a second implementation method of the function so as to generate a rebuilt query access plan". For the latter two limitations, the Examiner cites Lohman (U.S. Patent Application Publication No. 2002/0198867), and in particular, paragraphs [0106] and [0107].

Lohman (U.S. Patent Application Publication No. 2002/0198867), like Kabra, focuses on improving query optimization, in other words, improving the execution of sub-optimal queries. Lohman (U.S. Patent Application Publication No. 2002/0198867) accomplishes this by using empirical measurements from the query plan chosen for execution to validate if the model or estimates was correct or erroneous (e.g., claim 1 of Lohman (U.S. Patent Application Publication No. 2002/0198867)). If the model is in error, then one or more adjustments to the model are made to correct the error. However, like Kabra, the error in Lohman (U.S. Patent Application Publication No. 2002/0198867) has nothing to do with an error that halts execution; instead this error leads to sub-optimal plan selection. And although paragraphs [0106] and [0107] mention replacing a nested-loop join with a hash join, this is in the context of improving sub-optimal access plans, akin to Kabra, and does not disclose the remaining limitations in the same manner claimed.

Claim 7 is therefore at least patentable over the prior art for the reasons presented in connection with claim 1 above. Reconsideration and allowance of claim 7, and of claims 8 and 10-11 which depend therefrom, are therefore respectfully requested.

<u>Independent Claim 12</u>

Claim 12 generally recites a method for automatic handling of errors within a database engine. The method includes executing a query access plan comprising a plurality of functions, each function including a first implementation method, and the query access plan of the type generated by a query optimizer, detecting a first error when executing a first function. The first error is an execution error of a type that halts execution of the query access plan. The method also includes rebuilding the query access plan with the query optimizer to generate a new query access plan, executing the new query access plan to generate at least a portion of a result set for storage or display, receiving a second error while executing the first function within the new query access plan, and rebuilding the new query access plan by replacing the first implementation method with a second implementation method of the function.

In rejecting claim 12, the Examiner also relies on the same arguments and citations of Kabra, Brundage, and Lohman (U.S. Patent Application Publication No. 2002/0198867) made in connection with claims 1 and 7. The Examiner also admits that this combination does not disclose that a second error occurs. For this limitation, the Examiner states that "it would be obvious to one of ordinary skill in the art to apply the same steps to utilized[sic] to re-optimize the first sub-optimal plan to optimize a second sub-optimal plan since this is merely a repetitive step that occurs repetitively with any query plan."

Claim 12 is at least patentable over the prior art for the reasons presented in connection with claims 1 and 7 above. In particular, none of Kabra, Brundage and Lohman discloses or suggests rebuilding a query access plan with a query optimizer to generate a new query access plan and executing the new query access plan to generate at least a portion of a result set for storage or display, in response to an execution error of the type that halts execution of a query. Moreover, none of the references addresses any functionality for handling a subsequent error that occurs after a query access plan has been rebuilt and re-executed. The Examiner merely discounts this additional functionality, but the fact remains that none of the references discloses or suggests a multi-level response strategy that attempts to address execution errors in the hierarchical fashion recited in claim 12. As is even acknowledged by Kabra in connection with handling sub-optimal queries, re-optimizing queries comes with a performance cost, so the costs of any error handling processes must be weighed against potential benefits. Claim 12 is directed to a process whereby different operations are performed the second time an error is received than a first time, with the expectation that errors can be handled in an efficient and competent manner. None of the art cited by the Examiner appreciates the desirability of such a multi-step approach. Accordingly, Applicant submits that claim 12 is additionally patentable over the cited references for this additional reason.

Applicant accordingly submits that the combination proposed by the Examiner does not disclose or suggest each and every limitation in claim 12, and therefore, Applicant respectfully submits that independent claim 12 is non-obvious over Kabra, Brundage, and Lohman (U.S. Patent Application Publication No. 2002/0198867). Reversal of the Examiner's rejection of claim 12, and allowance of the claim, are accordingly requested.

Dependent Claims 4 and 10

Claims 4 and 10, which depend from claims 1 and 7, respectively, recite to varying extents the concept of handling another error detected while executing a query access plan that has been automatically rebuilt in response to an execution error by rebuilding the query access plan to replace a first implementation method of a function with a second implementation method. Claim 4 is representative, and recites receiving another error while executing a function within the new query access plan, identifying a first implementation method of the function within the new query access plan, and rebuilding the new query access plan by replacing the first implementation method with a second implementation method of the function so as to generate a rebuilt query access plan. Claim 10 adds the concept of replacing a second implementation of a function with a third implementation in response to an additional error.

In rejecting claim 4, the Examiner relies on paragraphs [0106] and [0107] of Lohman (U.S. Patent Application Publication No. 2002/0198867). In addition, as noted above in connection with claim 12, irrespective of whether any of the operations disclosed in Kabra, Brundage, and Lohman (U.S. Patent Application Publication No. 2002/0198867) can be analogized to functions or function implementations, none of the references discloses a multi-stage error handling protocol capable of handling multiple errors that are encountered during the execution of a query plan. Reversal of the Examiner's rejections of claims 4 and 10, and allowance of those claims, are therefore respectfully requested.

Dependent Claims 5 and 11

Claims 5 and claim 11, which depend from claims 1 and 7, respectively, recite to varying extents logging information about at least one error and the new query access plan. Claim 11 is representative, and recites logging information about the error, the another error, and the new query access plan.

In rejecting claim 11, Examiner relies on paragraph [0071] of Lohman (U.S. Patent Application Publication No. 2002/0198867). However, paragraph [0071] of Lohman refers to an old estimate, a new estimate, etc. The passage does not disclose logging information about errors and the new query access plan. Logging estimates or statistics is NOT the same as logging information about errors and the query access plan. The cited references therefore fail to disclose or suggest the additional features recited in claims 5 and 11. Reversal of the Examiner's rejections of these claims, and allowance of these claims, are therefore respectfully requested.

Dependent Claim 8

Claim 8 is not argued separately.

C. Conclusion

Applicant respectfully requests that the Board reverse the Examiner's rejections of claims 1, 3-8, and 10-12, and that the Application be passed to issue. If there are any questions regarding the foregoing, please contact the undersigned at 513/241-2324. If any other charges or credits are necessary to complete this communication, please apply them to Deposit Account 23-3000.

Respectfully submitted,

August 17, 2009
Date

/Scott A. Stinebruner/
Scott A. Stinebruner
Reg. No. 38,323
WOOD, HERRON & EVANS, L.L.P.
2700 Carew Tower
441 Vine Street
Cincinnati, Ohio 45202
Telephone: (513) 241-2324
Facsimile: (513) 241-6234

VIII. CLAIMS APPENDIX: CLAIMS ON APPEAL (S/N 10/754,010)

Listing of Claims:

1. (Previously Presented) A method for automatic handling of errors within a database engine, the method comprising the steps of:

detecting an error while executing a query access plan, wherein the error is an execution error of a type that halts execution of the query access plan, and wherein the query access plan is of the type generated by a query optimizer;

in response to detecting the error, automatically rebuilding the query access plan with the query optimizer to generate a new query access plan; and

executing the new query access plan to generate at least a portion of a result set for storage or display.

2. (Cancelled)

3. (Original) The method of claim 1, wherein the error is a function check.

4. (Original) The method of claim 1 further comprising the steps of:

receiving another error while executing a function within the new query access plan;

identifying a first implementation method of the function within the new query access plan; and

rebuilding the new query access plan by replacing the first implementation method with a second implementation method of the function so as to generate a rebuilt query access plan.

5. (Original) The method according to claim 1, further comprising the step of:

logging information about the error, and the new query access plan.

6. (Original) The method according to claim 1, further comprising the step of:

reporting the error.

7. (Previously Presented) A method for automatic handling of errors within a database engine, the method comprising the steps of:

receiving an error while executing a function within a query access plan, wherein the error is an execution error of a type that halts execution of the query access plan, and wherein the query access plan is of the type generated by a query optimizer;

identifying a first implementation method of the function within the query access plan;

rebuilding the query access plan with the query optimizer by replacing the first implementation method with a second implementation method of the function so as to generate a new query access plan; and

executing the new query access plan to generate at least a portion of a result set for storage or display.

8. (Original) The method of claim 7, wherein the function is one of a join function, an indexing function, a grouping function, and an ordering function.

9. (Cancelled)

10. (Previously Presented) The method of claim 7, further comprising the steps of:

receiving another error while executing the function within the new query access plan; and

rebuilding the new query access plan by replacing the second implementation method with a third implementation method of the function.

11. (Original) The method according to claim 10 further comprising the step of:

logging information about the error, the another error, and the new query access plan.

12. (Previously Presented) A method for automatic handling of errors within a database engine, the method comprising the steps of:

executing a query access plan comprising a plurality of functions, each function including a first implementation method, and the query access plan of the type generated by a query optimizer;

detecting a first error when executing a first function, wherein the first error is an execution error of a type that halts execution of the query access plan;

rebuilding the query access plan with the query optimizer to generate a new query access plan;

executing the new query access plan to generate at least a portion of a result set for storage or display;

receiving a second error while executing the first function within the new query access plan; and

rebuilding the new query access plan by replacing the first implementation method with a second implementation method of the function.

13.- 21 (Cancelled)

IX. EVIDENCE APPENDIX

[APPLICATION NO. 10/754,010]

None.

X. <u>RELATED PROCEEDINGS APPENDIX</u>

<u>[APPLICATION NO. 10/754,010]</u>

None.